# PART7　論理演算子と関数

目標
- 論理演算子の使い方を理解する。
- 関数を使ったプログラムを書けるようになる。

## Chapter1　論理演算子と関数

### 論理（Bool）値と論理演算子

論理値の変数には True もしくは False の値のみ代入できる。

| | | |
|---|---|---|
| **論理積** | and | and の前後の両方が True であれば True |
| **論理和** | or | or の前後のどちらか片方でも True であれば True |
| **否定** | not | 論理値（True・False）を反転<br>例）True を False に，False を True に |

```
# 論理演算子 and, or, not の使い方
x = 3
print(x > 2 and x < 4)
print(x < 2 or x > 4)
print(not(x < 2))
```

実行結果

```
True
False
True
```

覚えておこう

| 比較演算子 | 意味 |
|---|---|
| == | 等しい |
| != | 等しくない |
| > | より大きい |
| < | より小さい |
| >= | 以上 |
| <= | 以下 |

### 関数の定義

プログラムでは，構造を分かりやすくするため，〔①　　　　〕という形で，コードのまとまりに名前を付け，他から呼び出し可能な構造にすることがある。

関数は次のように定義する。

```
def 関数名 ( 引数 ):
  関数内での処理（複数行可）
  return 戻り値
```

関数の中で値を受け取るように指定された変数を〔②　　　　　〕とよぶ。〔③　　　　　〕で，関数内で処理をした結果の値を，呼び出し元に〔④　　　　　〕として受け渡すことができる。

## 引数として入力された値の２倍を計算する関数

```
def baigaeshi(x):
  y = 2 * x
  return y
```

```
print(baigaeshi(3))
```

実行結果

```
6
```

## 引数として入力された値が３の倍数かどうかを判定する関数

ｎの倍数を判定するプログラムは，ｎで割った余りが０である条件を利用する。

```
def baisu3(x):
  y = x % 3 == 0
  return y
```

```
print(baisu3(3))
print(baisu3(5))
```

実行結果

```
True
False
```

覚えておこう

関数の仕組み（要約）
- 引数で値を受け取る。
- return で戻り値を受け渡す。

知っておこう

引数は英語では parameter といい，直訳すると「渡し値」である。戻り値は英語では return value といい，直訳すると「返し値」である。

## 引数として入力された値が３と５の両方で割り切れるか判定する関数

```
def baisu35(x):
  y = x % 3 == 0 and x % 5 == 0
  return y
```

```
print(baisu35(15))
```

実行結果

```
True
```

## 引数と戻り値が配列の関数

- 引数として与えられた配列の要素の値が５の倍数かをそれぞれ判定する関数

```
def baisu5(X):
  n = len(X)
  for i in range(n):
    X[i] = X[i] % 5 == 0
  return X
```

```
Z = [1,2,3,4,5,6,7,8,9,10]
print(Z)
baisu5(Z)
```

実行結果

```
[1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
[False, False, False, False, True, False, False, False, False, True]
```

### • バブルソート「bubblesort」関数

バブルソートとは，並び替え（ソート）のアルゴリズムのひとつで，隣り合うデータを順番に，1 つずつ大きさを比較・序列する方式のことである。

```
# バブルソート（昇順）
def bubblesort(X):
  n = len(X)
  for i in range(n):
    for j in range(n - i - 1):
      if X[j] > X[j + 1]: # 隣り合う要素を比較
        temp = X[j] # 隣り合う要素を交換
        X[j] = X[j + 1]
        X[j + 1] = temp
  return X
```

```
A = [6,1,8,3,5]
bubblesort(A)
```

実行結果

```
[1, 3, 5, 6, 8]
```

まとめ

・基本的な**論理演算子**として，次の 3 つがある。

| | | |
|---|---|---|
| **論理積** | and | 両方が True であれば True |
| **論理和** | or | どちらか片方でも True であれば True |
| **否定** | not | 論理値（True・False）を反転<br>例）True を False に，False を True |

・**関数**は def 構文を使って定義する。

```
def 関数名 ( 引数 1):
  関数内での処理
  return 戻り値
```

# Chapter2　実践問題

問 1　次の Python プログラムは，引数として入力された値が 2 または 3 で割り切れるか判定する関数 baisu23 である。空欄 ( ア )( イ ) を穴埋めしなさい。

```
def baisu23(x):
  y = x % 2 == 0 [ (ア) ] x % 3 == 0
  [ (イ) ] y
```

```
print(baisu23(4))
print(baisu23(11))
print(baisu23(15))
```

実行結果

```
True
False
True
```

問 2

(1)　次の Python プログラムは数が 5 で割り切れれば Buzz と表示させ，その他の数はそのままの数を表示する buzz 関数である。空欄 ( ア )( イ ) を穴埋めしなさい。

```
def buzz(i):
  if i % 5 == [ (ア) ]:
    x = 'Buzz'
  else:
    x = [ (イ) ]
  return x
```

```
print(buzz(5))
print(buzz(8))
```

実行結果

```
buzz
8
```

(2)　次の Python プログラムは下記の機能を満たす fizzbuzz 関数である。

1. 数が 3 で割り切れれば，代わりに Fizz と表示する。
2. 数が 5 で割り切れれば，代わりに Buzz と表示する。
3. 数が 3 と 5 の両方で割り切れれば，代わりに FizzBuzz と表示する。
4. 1 ～ 3 の条件を満たさない場合はそのままの数を表示する。

空欄 ( ウ )( エ ) を穴埋めしなさい。

```
def fizzbuzz(i):
  if i % 3 == 0 (ウ) i % 5 == 0:
    x = 'FizzBuzz'
  (エ) i % 3 == 0:
    x = 'Fizz'
  (エ) i % 5 == 0:
    x = 'Buzz'
  else:
    x = i
  return x
```

```
print(fizzbuzz(6))
print(fizzbuzz(10))
print(fizzbuzz(30))
print(fizzbuzz(17))
```

実行結果

```
Fizz
Buzz
FizzBuzz
17
```

**問 3**　次の Python プログラムは，2 つの整数値の要素からなる配列 Z を入力したときに，要素の値を昇順にした配列を出力する関数 bubblesort2 である。空欄 ( ア )( イ ) を穴埋めしなさい。

```
def bubblesort2(X):
  if X[0] (ア) X[1]:
    temp = X[0]
    X[0] = X[1]
    X[1] = (イ)
  return(X)
```

```
Z = [2,1]
bubblesort2(Z)
```

実行結果

```
[1,2]
```

# 第 6 講　PART7　確認問題

問 1　論理演算子についてまとめた次の図について，次の各問いに答えなさい。

| 論理演算子 | | |
|---|---|---|
| 論理積 | （ア） | 両方が True であれば True |
| 論理和 | （イ） | どちらか片方でも True であれば True |
| 否定 | （ウ） | 論理値（True・False）を反転 |

(1)　空欄（ア）に入るものとして最も適切なものを，次の①～④のうちから一つ選びなさい。

①　and　　②　nor
③　not　　④　or

(2)　空欄（イ）に入るものとして最も適切なものを，次の①～④のうちから一つ選びなさい。

①　and　　②　nor
③　not　　④　or

(3)　空欄（ウ）に入るものとして最も適切なものを，次の①～④のうちから一つ選びなさい。

①　and　　②　nor
③　not　　④　or

問 2　Python で関数を定義するときの記述の仕方について，次の各問いに答えなさい。

```
def 関数名 ( （ア） ):
  関数内での処理
  return （イ）
```

(1) 空欄（ア）に入るものとして最も適切なものを，次の①～④のうちから一つ選びなさい。

① 定数　② 引数
③ 補数　④ 戻り値

(2) 空欄（イ）に入るものとして最も適切なものを，次の①～④のうちから一つ選びなさい。

① 定数　② 引数
③ 補数　④ 戻り値

**問３** 次の Python プログラムの実行結果について，最も適切なものを，①～④のうちから一つ選びなさい。

```
def plus7(x):
  s = x + 7
  return s

print(plus7(5))
```

① 5　② 7
③ 12　④ 2

# 第６講　アルゴリズムとプログラミング

## PART1　アルゴリズムの表現方法

問１　（ア）②　（イ）①　（ウ）③　（エ）④　（オ）⑤

問２　（ア）③　（イ）②　（ウ）①　（エ）⑤　（オ）④

問３　③

## PART2　プログラミングの基本

問１　(1) 省略　(2) 省略　(3) 省略

## PART3　ネットワークを利用したプログラミング

問１　(1) 省略　(2) 省略

## PART4　変数の型と関数

問１　(1) `print(a // b)`

(2) `print(a % b)`

(3) `print(a ** b)`

問２

```
a = 'shibuya'
b = 109
print(a + str(b))
```

問３　`print(a + str(x) + '歳です')`

問４　`print(float(x) / y)`

## PART5　モジュールとfor文・if文

問１　（ア）math　（イ）math.pi

問２　（ア）random　（イ）random.randint

問３　（ア）(10,15)　（イ）a == 10:

問４　（ア）range(4)　（イ）cnt == 0

## PART6　配列と反復処理

問１　（ア）len(A)　（イ）A[i]

問２　（ア）temp　（イ）X[i]

問３　（ア）n　（イ）r

問４　（ア）Coin[i]　（イ）%

## PART7　論理演算子と関数

問１　（ア）`or`　（イ）`return`

問２　(1)（ア）`0`　（イ）`i`

(2)（ウ）`and`　（エ）`elif`

問３　（ア）`>`　（イ）`temp`

## PART8　【発展】Web APIや外部データの活用

問１　（ア）`requests`　（イ）`'address1'`

問２　（ア）`pandas`　（イ）`matplotlib.pyplot`

（ウ）'昨日までの観測史上１位の値（℃）'

# 第 6 講　PART7　確認問題　解答

問 1

(1) ①

解説：

両方が True であれば True を返すのは and である。

(2) ④

解説：

どちらか片方でも True であれば True を返すのは or である。

(3) ③

解説：

論理値（True・False）を反転するのは not である。

問 2

(1) ②

解説：

関数が呼び出されたときに値を受け取る変数を引数という。

(2) ④

解説：

関数を実行した際，return によって呼び出し元に返す値を戻り値という。

問 3　③

解説：

このプログラムに記述されている plus7 関数は引数に 7 を足した結果を呼び出し元に返している。

5 行目で 5 を引数に呼び出された plus7 関数は，5+7 の結果を変数 s（12）に代入して呼び出し元に返す。その値が print 関数によって画面に表示されるので，正解は 12 である。